What is claimed is:

1. A method for protecting an application from executing an illegal or harmful operation request received from a distrusted environment, the method comprising the steps of:

determining whether said operation request is illegal or harmful to an environment of said application, and

preventing an application from executing an illegal or harmful operation request.

2. The method of claim 1, wherein said illegal and harmful operation request causes damage or allows unauthorized access to a computer system element or parameter selected from the group consisting of: said application, an application other than said application, a trusted network, one or more data files, memory, buffers, performance, confidential information, hardware, software, a database, an information server, and any combination thereof.

3. The method of claim 1, wherein said illegal and harmful operation request is selected from the group consisting of: database manipulation, Internet information server vulnerability, application vulnerability, URL manipulation, business process manipulation, poisoning attack, and any combination thereof.

4. The method of claim 1, wherein said step of preventing comprises the step of

rejecting said illegal or harmful operation request.

5. The method of claim 1, wherein said step of preventing comprises the step of

modifying said illegal or harmful operation request into a legal or harmless operation request.

6. The method of claim 1, wherein said step of preventing comprises the step of

replacing said illegal or harmful operation request with a legal or harmless operation request.

7. The method of claim 1, wherein said step of determining comprises the step of

checking said operation request for an existence of an embedded command causing database manipulation.

8. The method of claim 5, wherein said embedded command is an SQL based command.

9. The method of claim 7, further comprising the steps of:

parsing said operation request into one or more expressions;

building a state-automate;

inspecting said one or more expressions for improper syntax and characters not defined in

a first alphabet; and

applying said state-automate to said operation request.

10. The method of claim 9, wherein said alphabet is selected from the group consisting of: letters, digits, and encoded characters; blocks of letters; groups of said blocks; and any combination thereof.

11. The method of claim 1, wherein said step of determining comprises the steps of:

comparing said operation request against stored known vulnerability patterns to determine a match; and

blocking said operation request if said match is found.

12. The method of claim 11, wherein said step of determining further comprises the step of:

updating said stored vulnerability patterns with newly found vulnerability patterns.

13. The method of claim 11, wherein said step of comparing comprises the steps of:

converting every consecutive specified number of characters in said operation request into n-bits of binary code;

computing a hash value for said every consecutive specified number of characters in said operation request;

comparing every hash value to stored hash values representing vulnerability patterns.

14. The method of claim 13, wherein said n-bits is 8-bits and said specified number is equal to four.

15. The method of claim 1, wherein said step of determining comprises the steps of:

dividing said operation request into four zones;

comparing each of said four zones against stored known vulnerability patterns to determine a match; and

blocking said operation request if said match is found.

16. The method of claim 15, wherein said four zones represent a URI, query string, header, and body associated with said operation request.

17. The method of claim 1, further comprising the steps of:

sending legal or harmless operation requests to said operation; and

generating a reply to said operation request.

18. The method of claim 17, wherein said step of determining further comprises the steps of:

determining a first set of internal URLs and parameters values contained in said reply;

receiving a second operation request in response to said reply;

comparing a second set of internal URLs and parameters values contained in said second operation request with said first set to determine if said sets correspond; and

rejecting said second operation request if said sets do not correspond.

19. The method of claim 17, further comprising the steps of:

identifying a single client to interact with said application;

determining a first set of parameter names and values in said reply;

receiving a second operation request from said client in response to said reply;

determining a second set of parameter names and values in said second operation request; and

forwarding said second request to said application only if said first set matches said second set.

20. The method of claim 1, further comprising the steps of:

designating an application path of said application as restricted;

determining a destination of said operation request; and

blocking said operation request if said destination is equal to said designated application path.

21. The method of claim 1, wherein said step of determining comprises the steps of:

compiling a list of acceptable operation requests; and

comparing said operation request to said list of acceptable operation requests.

22. The method of claim 1, wherein said step of determining comprises the steps of:

determining a parameter value contained within said operation request; and

applying a pre-defined rule to said parameter based on said parameter type, wherein said pre-defined rule defines one or more acceptable parameter values.

23. The method of claim 1, wherein said step of determining comprises the steps of:

identifying a cookie message header in said operation request;

decrypting values in said cookie message header; and

modifying said operation request to reflect said decrypted values.

24. The method of claim 17, further comprising the steps of:

identifying a cookie message header in said reply;

encrypting values in said cookie message header; and

modifying said reply to reflect said encrypted values.

25. The method of claim 1, further comprising the steps of:

receiving a plurality of operation requests;

storing said plurality of operation requests into a virtual directory;

building a dynamic range of entered values for each parameter in said plurality of operation requests;

computing an acceptable range of values for each parameter based on a statistical model applied to said dynamic range of entered values for each value;

receiving a subsequent operation request;

identifying parameter values in said subsequent operation request; and

determining if said parameter values in said subsequent operation request are within said acceptable range of values.

26. The method of claim 25, further comprising the steps of:

adding said parameter values in subsequent operation request to dynamic range;

adjusting said acceptable range of values for each parameter by applying said statistical model.

27. A method for preventing one or more applications from executing out of their intended scopes of operation, comprising the steps of:

receiving one or more operation requests;

formatting each operation request into a formatted message according to a designated communications protocol, wherein said designation communication protocol is determined by the type of application being requested;

indexing said one or more formatted messages;

storing a copy of said indexed one or more formatted messages;

translating said formatted messages into internal messages according to an encoding scheme;

resolving a destination node for each operation request; and

applying one or more security pipes to each operation request, wherein the number and types of pipes applied to each operation request are based on said resolved destination node of each operation request.

28. The method of claim 27, wherein said designated communications protocols are selected from the group consisting of:

HTTP, HTTPs, HTTPd, WebDAV, and SOAP.

29. The method of claim 27, wherein said encoding scheme is ASCII.

30. The method of claim 27, wherein application of a pipe comprises the steps of:

parsing a first operation request into one or more expressions;

building a state-automate;

inspecting said one or more expressions for improper syntax and characters not defined in a first alphabet; and

applying said state-automate to said first operation request.

31. The method of claim 30, wherein said alphabet is selected from the group consisting of: letters, digits, and encoded characters; blocks of letters; groups of said blocks; and any combination thereof.

32. The method of claim 27, wherein application of a pipe comprises the steps of

comparing said operation request against stored known vulnerability patterns to determine a match; and

blocking said operation request if said match is found.

33. The method of claim 32, further comprising the step of:

updating said stored vulnerability patterns with newly found vulnerability patterns.

34. The method of claim 27, wherein said step of comparing comprises the steps of:

converting every consecutive specified number of characters in said operation request into n-bits of binary code;

computing a hash value for said every consecutive specified number of characters in said operation request;

comparing every hash value to stored hash values representing vulnerability patterns.

35. The method of claim 34, wherein said n-bits is 8-bits and said specified number is equal to four.

36. The method of claim 27, wherein application of a pipe comprises the steps of

dividing said operation request into four zones;

comparing each of said four zones against stored known vulnerability patterns to

determine a match; and

blocking said operation request if said match is found.

37. The method of claim 36, wherein said four zones represent a URI, query string, header, and body associated with said operation request.

38. The method of claim 27, further comprising the steps of:

sending legal or harmless operation requests to said operation; and

generating a reply to said operation request.

39. The method of claim 38, wherein application of a pipe comprises the steps of:

determining a first set of internal URLs and parameters values contained in said reply;

receiving a second operation request in response to said reply;

comparing a second set of internal URLs and parameters values contained in said second operation request with said first set to determine if said sets correspond; and

rejecting said second operation request if said sets do not correspond.

40. The method of claim 38, wherein application of a pipe comprises the steps of:

identifying a single client to interact with said application;

determining a first set of parameter names and values in said reply;

receiving a second operation request from said client in response to said reply;

determining a second set of parameter names and values in said second operation request; and

forwarding said second request to said application only if said first set matches said second set.

41. The method of claim 27, wherein application of a pipe comprises the steps of:

designating an application path of said application as restricted;

determining a destination of said operation request; and

blocking said operation request if said destination is equal to said designated application path.

42. The method of claim 27, wherein application of a pipe comprises the steps of:

compiling a list of acceptable operation requests; and

comparing said operation request to said list of acceptable operation requests.

43. The method of claim 27, wherein application of a pipe comprises the steps of:

determining a parameter value contained within said operation request; and

applying a pre-defined rule to said parameter based on said parameter type, wherein said pre-defined rule defines one or more acceptable parameter values.

44. The method of claim 27, wherein application of a pipe comprises the steps of:

identifying a cookie message header in said operation request;

decrypting values in said cookie message header; and

modifying said operation request to reflect said decrypted values.

45. The method of claim 27, wherein application of a pipe comprises the steps of:

identifying a cookie message header in said reply;

encrypting values in said cookie message header; and

modifying said reply to reflect said encrypted values.

46. The method of claim 27, wherein application of a pipe comprises the steps of:

storing said plurality of operation requests into a virtual directory;

building a dynamic range of entered values for each parameter in said plurality of operation requests;

computing an acceptable range of values for each parameter based on a statistical model applied to said dynamic range of entered values for each value;

receiving a subsequent operation request;

identifying parameter values in said subsequent operation request; and

determining if said parameter values in said subsequent operation request are within said acceptable range of values.

47. The method of claim 46, further comprising the steps of:

adding said parameter values in subsequent operation request to dynamic range;

adjusting said acceptable range of values for each parameter by applying said statistical model.

48. A system for implement an application security layer between a trusted application and a distrusted computer environments comprising:

means for receiving an operation request for said application;

means for embedding said operation request into a data format used by said trusted application; and

means for checking a contents of said operation request to identify if said operation request is illegal or harmful to an environment of said application, wherein said consists of a

message header, message body, query string, encoding type, and an uniform resource identifier ("URI").

49. The system of claim 48, wherein said data format is selected from the group consisting of:

HTTP, HTTPs, HTTPd, WebDAV, and SOAP.

50. The system of claim 48, wherein said receiving means is a queued socket server.

51. A system for implement an application security layer between a trusted application and a distrusted computer environments comprising:

means for receiving an operation request for said application;

means for embedding said operation request into a data format used by said trusted application;

an encoder for encoding said operation request according to an encoding scheme; and

means for applying one or more security pipes to each operation request, wherein the number and types of pipes applied to each operation request are based on said resolved destination node of each operation request.

52. The system of claim 51, wherein said data format is selected from the group consisting of:

HTTP, HTTPs, HTTPd, WebDAV, and SOAP.

53. The system of claim 51, wherein said encoding scheme is ASCII.

54. The system of claim 51, further comprising

means for providing a firewall.